第3章

信号処理システム設計を体験する

――シミュレータで設定を変えながらディジタルPLLの動作を解析

ルーウィ ヴァレニャ

ここでは，同期検波用の全ディジタルPLL（phase-locked loop）を例に，評価版のシミュレータを使いながら，信号処理システムの動作を確認する．PLLは，通信回路やクロック回路などに利用される重要な回路である．シミュレータを使うことで，PLLのさまざまな動作を視覚的に確認することができる．なお，本稿で説明する作業に必要な評価版ツールや設計データは，すべて本誌付属のCD-ROMに収録されている．（編集部）

無線通信システムでは，送信側と受信側が空間的に離れています．そのため，ほとんどの場合，使っている基準クロックは独立しており，周波数が一致していません注1．送信側と受信側の周波数（位相）を正確に合わせる必要がない応用もありますが（例えば，低コストのAMラジオなど），無線LAN（local area network）や携帯電話などでは周波数を合わせるために同期検波がよく使われます．同期検波の

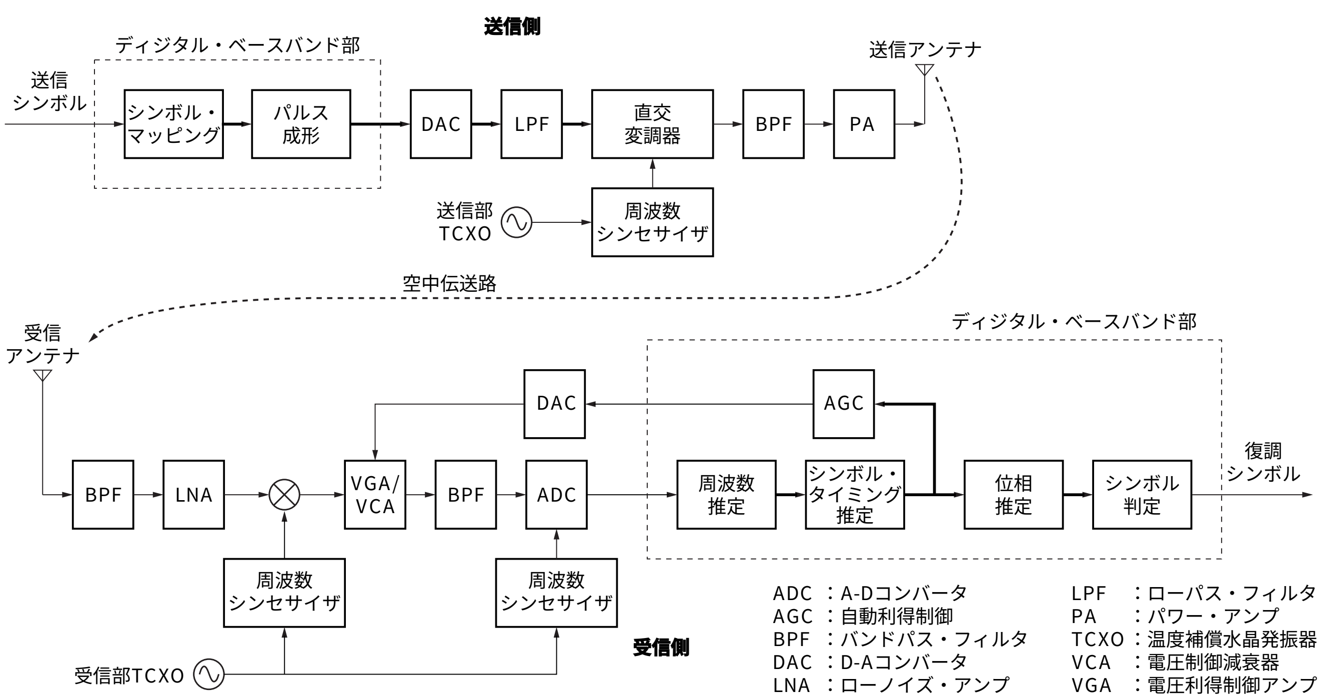


図1　無線通信システムの概要

太い矢印は複素信号を表す．送信側と受信側が利用している基準クロック（TCXO）は完全に独立しているため，まったく同じ周波数にはならない．同期検波では，受信側の搬送波の周波数と位相を送信側に合わさなければならない．周波数推定部と位相推定部がその役目を果たす．位相推定の一つの実装例がPLL（phase-locked loop）である．

注1：例えば，送信側と受信側がGPS信号を利用すれば，周波数を合わせられる．しかし，応用によってはドップラ・シフトが生じ，受信側がGPS信号から生成したリファレンスの周波数と受信した搬送波の周波数が異なる場合がある．また，伝達遅延によって位相が変わることもある．送受信双方が利用しているタイミング基準（例えば，温度補償水晶発振器）の周波数の標準値は同じであっても，温度，電源電圧，湿度などによって周波数誤差が生じることもある．

一つの実装例がPLL（phase-locked loop）です．一般的な無線通信システムの構成を**図1**に示します．

このような複雑なシステムを正しく理解し，解析し，設計するには，シミュレーションは欠かせない手法です．本稿では，シミュレータを使って動作を解析しながら，同期検波用PLLについて勉強していきます．

● 信号処理シミュレータをインストールする

本誌付属のCD-ROMには，Windows XP上で動作する米国CoWare社のディジタル信号処理システム開発ツール「SPW 5-XP（評価版）」が収録されています．これを実際に動かしながら，本稿を読まれることをお勧めします．

本ツールをインストールするにあたって，評価用ライセンスを取得する必要があります．まず，コーウェア（CoWare社の日本法人）のホームページ（http://www.coware.co.jp/）を開いて，「SPW 5-XP無償評価プログラム」のところをクリックします．そして，必須情報を記入し，［評価希望］ボタンを押します．HOST IDを聞かれますが，これはインストールするパソコンのMAC（media access control）アドレスです．MACアドレスを調べるには，Windowsのコマンド・プロンプト（DOS窓）でipconfig/allコマンドを実行します．「Ethernet adapter ローカル エリア接続」の「Physical Address」の部分がMACアドレスになります．3営業日中に30日間の評価ライセンスが電子メールで届きます．ライセンス・ファイルは適当な場所（「マイ ドキュメント」フォルダなど）に保存します．

本ツールをインストールする際には，まず，米国Microsoft社の「Visual Studio .NET」をインストールしてから，付属CD-ROMに収録されているCoWare_SPW5-XP_Installer.exeを実行します[注2]．必要な情報を入力して，インストールを進めていきます．インストールが終わったら，C:\Documents and Settings\ユーザ名\MyDocuments\myworkspaceというフォルダが作成されます．これは本ツールのデフォルト・ワークスペースで，作成するファイルやシステムはこのフォルダに保存されます．付属CD-ROMの中のpll_demoフォルダに今回の例題に合わせたデモンストレーション・システムがあるので，myworkspaceフォルダにコピーしてください．

● シミュレーションを実行する

デスクトップにあるSPW 5-XPのアイコンをダブルクリックすると，ツールが起動します．ここで，左側にあるブラウザ領域（ファイル・ツリー形式）のUser Libsのpll_demoの下にあるsym_timing_and_freq_errの「system」をダブルクリックしてください．**図2**のような画面が表示されます．

注2：Visual Studio .NETを所有していないが，SPW 5-XPを評価したいという方は，Microsoft社の以下のサイトでVisual Studio .NETの60日間限定評価版を購入できる．https://s.microsoft.com/japan/ordercenter/developer/vs2003enteval.aspx

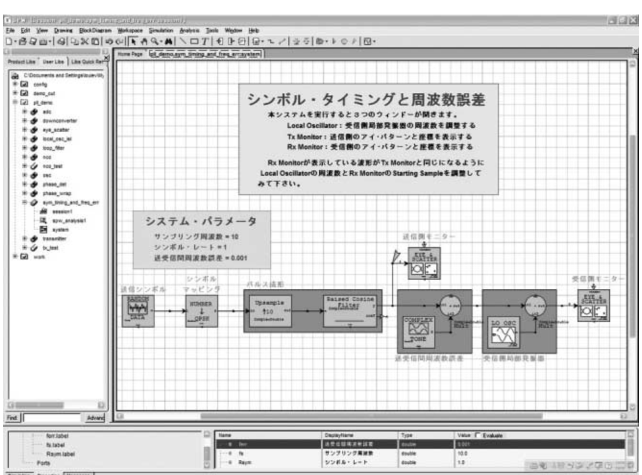


図2　SPW 5-XPの画面例

左側がファイル・ツリー形式のブラウザ領域，下が情報表示領域，中央（右上）がブロック図などを作成するデザイン領域になっている．システム・パラメータを変えると，そのパラメータを参照しているブロックも同時に変わる．ブロックをクリックすると，そのブロックのパラメータが情報ウィンドウに表示される．ダブルクリックすると，下の階層を開ける．最下層にあるブロックをダブルクリックすると，ソース・コードが表示される．

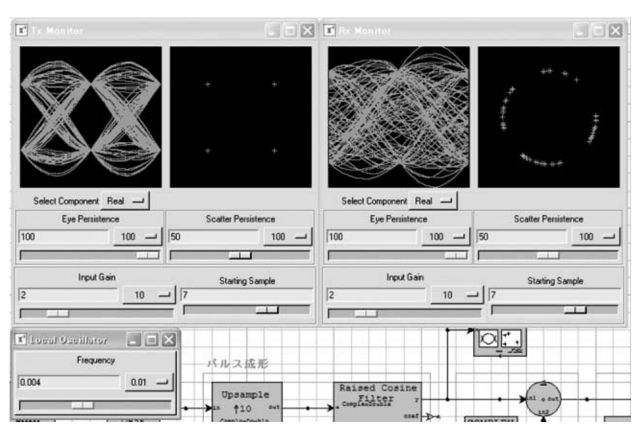


図3　シンボル・タイミングと周波数誤差

Tx Monitorの左にアイ・パターン（eye pattern）が，右に座標が表示されている．アイ・パターンとは，信号を時間軸上で決まった間隔で切って，重ね合わせた図である（ひし形の部分が“アイ”，つまり“目”）．本例ではQPSK（quadrature phase shift keying）が使われているので，四つの点が見える．この画面ではシンボル・タイミングはとれているが，周波数誤差が存在する．周波数誤差が残っていると，Rx Monitorの座標が回転しているように見える．

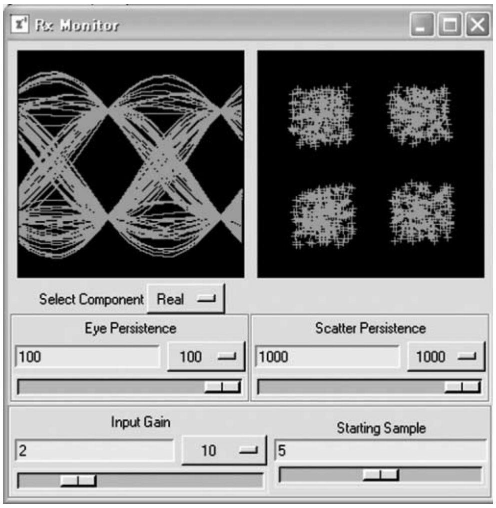


図4　周波数誤差はないがシンボル・タイミングが取れていない状態
シンボル点が広がっている．このような状態でも復調はできるが，ノイズが加わるとエラー率が高くなるため，望ましくない．

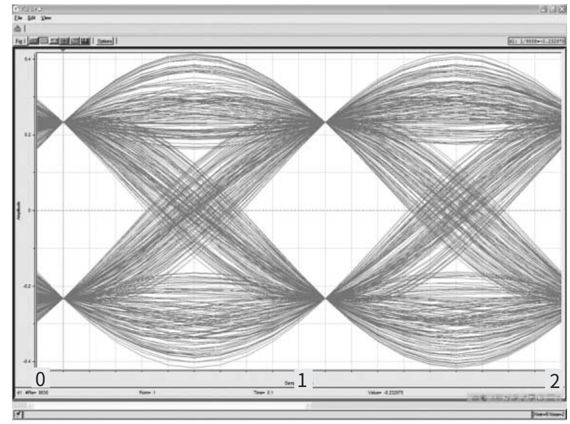


図5　アナリシスGUIで表示したアイ・パターン
本図でシンボル・レートを1MHzと仮定すると，サンプリング・レートは10MHzになる．つまり，各シンボルの間に9サンプルがある．最適なサンプル時刻は曲線が集中しているタイミング，または目がいちばん開いているタイミング，つまり，0.1，1.1，2.1，…である．これ以外のタイミングでサンプリングすると，**図4**のように座標が広がって見える．

メニュー・バーにある緑色の再生アイコンをクリックすると，シミュレーションが始まります．実行時の画面を**図3**に示します．送信側（搬送波に乗せる前の状態）と受信側のアイ・パターン，および座標が現れます．

Rx MonitorのStarting Sampleを変えると，シンボル・タイミングを調整できます．Local OscillatorのFrequencyを変えると，周波数誤差を調整できます．Rx Monitorの表示がTx Monitorと同じになるようにFrequencyとStarting Sampleを調整してみてください．シンボル・タイミングが合って周波数誤差が残っていると，**図3**のRx Monitorのように見えます．一方，周波数誤差がないものの，シンボル・タイミングが合っていない状態では**図4**のようになります．シンボル・タイミングとアイ・パターンについては**図5**を参照してください．実行を中止する場合は，メニュー・バーにある赤いSTOPアイコンをクリックします．

本記事では，シンボル・タイミングと周波数誤差は別々のブロックで推定し注3，シンボル・タイミングは合っているものと仮定します．

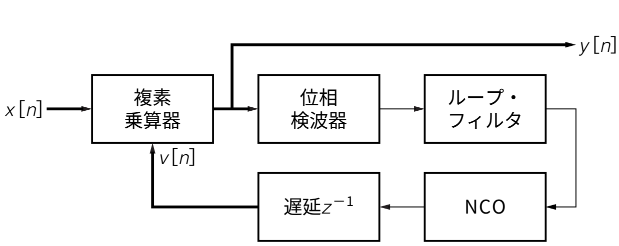


図6　同期検波用の全ディジタルPLLの構成
太い矢印は複素信号を表す．同期検波用の全ディジタルPLLは，複素乗算器（入力信号の位相を回転させる），位相検波器（入力信号の位相を算出する），ループ・フィルタ（PLL全体の特性を決める），NCO（数値制御発振器．入力の数値によって出力周波数が変わる），遅延（帰還があるため，NCOの値はすぐには出力されない．最低でも1サイクルの遅延が必要になる）から構成される．

● PLLとは？

ここで，PLLについて説明します．PLLをひと言で言うと，「自動的に周波数（位相）誤差をなくすための回路」です．つまり，二つの信号の位相を同期させる回路です．異なっている信号の相対位相が固定的な関係を持つように，片方の信号の位相を制御します．PLLは周波数シンセサイザ注4などによく使われますが，ここでは復調に使われる同期検波用のPLLに焦点を絞ります[1]．同期検波用のPLLの基本構成は**図6**のようになります．

全ディジタルPLL（フルディジタルPLL）は，文字どおりすべての要素にディジタル回路が使われているPLLです．実装例としては，専用ディジタル回路による構成と，汎用DSP（ディジタル信号処理プロセッサ）とソフトウェアによって実現する構成があります．昔は，同期検波に全アナログPLLがよく使われましたが，ディジタルLSI製造技術の進歩により，最近では全ディジタルPLLで実装されることが多くなりました．ただし，シンボル・レートが高くて（数

注3：本記事では説明しないが，同時に推定する方法（joint estimation）もある．
注4：周波数シンセサイザ（frequency synthesizer）とは，基準周波数から別の周波数を生成する回路．

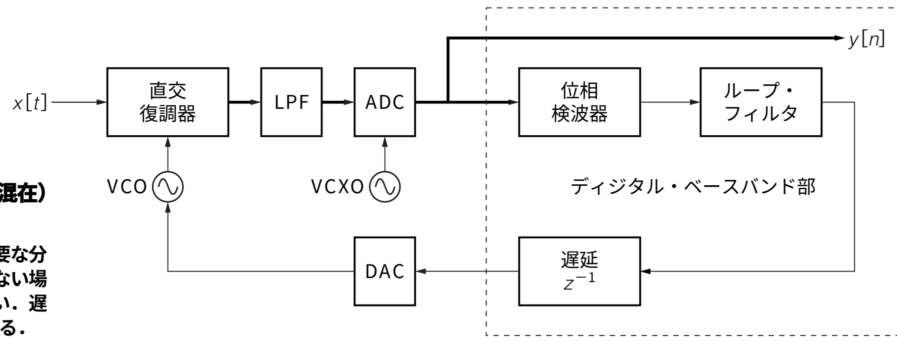


図7
ミックスト・シグナル（アナログ・ディジタル混在）の同期検波用PLL
太い矢印は複素信号を表す．A-Dコンバータに必要な分解能が高く，高速なクロックでサンプリングできない場合，ベースバンド部でサンプリングせざるを得ない．遅延部はD-Aコンバータ内部のレジスタで吸収できる．

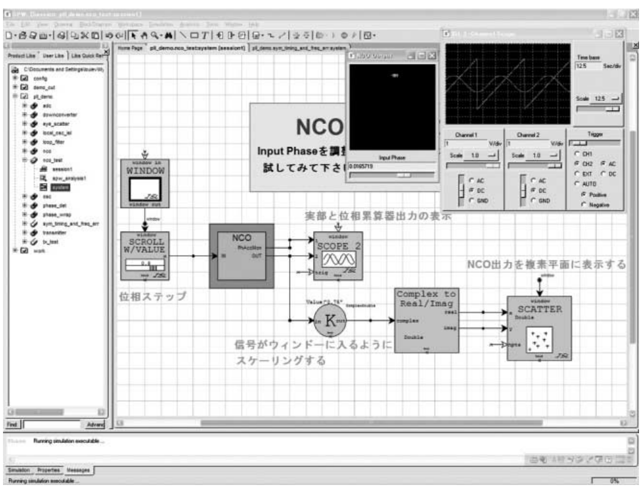


図8　NCOテストベンチと実行時の表示
オシロスコープのウィンドウに位相累算器とNCOの出力（実部）が表示されている．NCO Outputウィンドウには，NCO出力を複素平面で見たときの状態が示されている．Input Phaseのスライド・バーでNCOの入力位相を調整できる．波形を見やすくするため，オシロスコープ・ウィンドウのTime baseのScaleを「12.5」に，Channel 1，2をそれぞれ「DC」に，Triggerを「CH2」に設定する．ノコギリ波形は位相累算器の出力で，正弦波はNCO出力の実部．入力位相（Input Phase）でノコギリ波のスロープ（正弦波の周波数）を決める．入力位相が正数のとき，位相累算器の出力は－πから＋πに変わり，座標は反時計方向に回る．Input Phaseを調整してみて，出力波形がどう変わるか確認していただきたい．

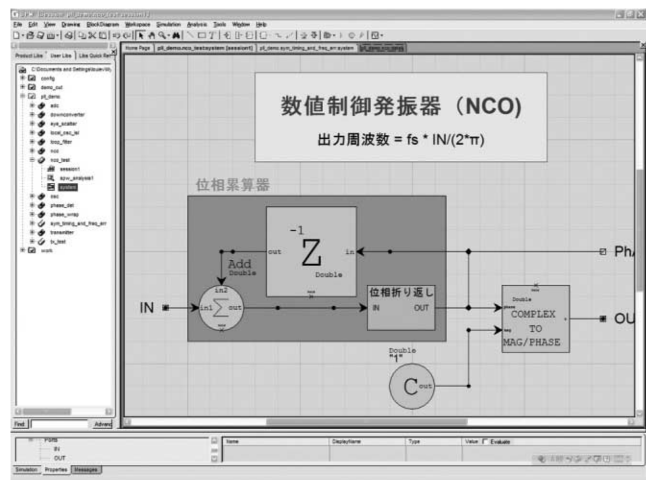


図9　NCOの詳細ビュー
NCOブロックをダブルクリックすると，この画面が表示される．位相累算器の周期性がよく見えるように，位相折り返しブロックを追加した．MAG/PHASE TO COMPLEXブロックについては，オイラーの公式（$e^{j\theta}=\cos\theta+j\sin\theta$）を利用して振幅と位相をsin，cosに変換する．

GHz)，A-Dコンバータのクロックが間に合わない場合には，アナログ系のPLLが採用されています(2)．**図7**に，ミックスト・シグナル（アナログ・ディジタル混在）PLLの構成例を示します．

最近の通信システムでは，エラー訂正や圧縮技術がよく採用されています．これらはディジタル回路で実装されているので，受信側で信号をディジタルに変換しなければなりません．LSI化を考えた場合，アナログ回路に比べてディジタル回路のほうがチップ面積を縮小できます．また，ディジタル回路には安定性が高く，ばらつきの影響を受けにくいという特徴があります．

● 全ディジタルPLLは五つの要素で構成される

ここでは全ディジタルPLLの各構成要素について，説明していきます．

1) NCO

NCO（numerically controlled oscillator；数値制御発振器）は複素トーン信号（$\cos 2\pi f_{NCO} nT_s + j\sin 2\pi f_{NCO} nT_s$）注5を生成します．NCOの動作を理解するため，pll_demoの下にあるnco_testの「system」を開き，実行してみてください．実行時の画面を**図8**に示します．

NCOブロックをダブルクリックすると，その詳細が表示されます（**図9**）．NCOは位相しか操作しません．NCOの伝達関数は次のようになります．

注5：f_{NCO}はトーンの周波数，T_sはサンプリング周期，nはサンプリング指数である．

3

$$
\begin{aligned}
OUT_{NCO}[n] &= \cos\left(IN_{NCO}[n] + OUT_{NCO}[n-1]\right) \\
&\quad + j\sin\left(IN_{NCO}[n] + OUT_{NCO}[n-1]\right) \\
&= \exp\left\{j\left(IN_{NCO}[n] + OUT_{NCO}[n-1]\right)\right\}
\end{aligned}
\tag{1}
$$

位相折り返しブロックは，位相累算器の出力を［−π，＋π］の範囲内にするために追加されています．加算器の出力がπを超えてπ＋θになると，位相折り返しの出力は−π＋θになります．このように，NCOの伝達関数には位相折り返しの影響はありません．実装上は位相折り返しの部分は不要ですが，これがないと位相累算器の周期性が見えないので，わかりやすくするために追加しました．

NCOの出力は複素トーンですが，その出力を複素平面で表すと**図10**のようになります．サンプル・レートf_sの速さと入力位相のステップで，信号点が回転します．信号点は2π分回ると1周します．例えば，入力位相が2π/16の場合，16サンプルで1周するので，出力トーンの周波数はf_s/16になります．したがって，NCOの出力トーンの周波数と入力位相の関係は次のようになります．

$$
f_{NCO} = \theta_{in} \frac{f_s}{2\pi} \tag{2}
$$

NCOの利得は，$K_{NCO} = f_s/2\pi$（Hz/rad）となります．

2）複素乗算器

複素乗算器は，入力信号を回転させます．**図6**を参照してください．入力信号$x[n]$は次のようになります．

$$
x[n] = A[n]\exp\left\{j\left(2\pi f_e n T_s + \phi[n]\right)\right\} \tag{3}
$$

ここで$A[n]$は入力信号の振幅変調分，$\phi[n]$は位相変調分，f_eは残留周波数誤差，T_sはサンプリング周期，nはサンプリング指数となります．式(1)と式(2)から，NCOの出力$v[n]$は次のようになります．

$$
v[n] = \exp\left(j2\pi f_{NCO} n T_s\right) \tag{4}
$$

複素乗算器の出力は，以下のようになります．

$$
\begin{aligned}
y[n] &= x[n] * v[n] \\
&= A[n]\exp\left\{j\left(2\pi f_e n T_s + \phi[n]\right)\right\}\exp\left(j2\pi f_{NCO} n T_s\right) \\
&= A[n]\exp\left\{j\left\{2\pi\left(f_e + f_{NCO}\right)nT_s + \phi[n]\right\}\right\} \\
&= A[n]\exp\left\{j\phi[n]\right\}, \quad \text{when} f_{NCO} = -f_e
\end{aligned}
\tag{5}
$$

PLLは，$f_{NCO} = -f_e$になるように動作します．

3）位相検波器

PLLの入力信号は複素信号ですが，観測しているパラメータはその信号の位相です．位相検波器は，入力信号の位相を算出する回路です．基本的にarctangent関数を利用しますが，入力信号の変調方式によって多少変更する必要があります（**図11**）．

$$
OUT_{PD}[n] = \tan^{-1}\left(\frac{IN_{imag}}{IN_{real}}\right) \qquad \text{CW入力時} \tag{6a}
$$

$$
OUT_{PD}[n] = \begin{cases}
\tan^{-1}\left(\dfrac{IN_{imag}}{IN_{real}}\right), & -\dfrac{\pi}{2} \leqq \theta_{in} < \dfrac{\pi}{2} \\
\pi + \tan^{-1}\left(\dfrac{IN_{imag}}{IN_{real}}\right), & \dfrac{\pi}{2} \leqq \theta_{in} < -\dfrac{\pi}{2}
\end{cases}
\tag{6b}
$$

BPSK入力時

$$
OUT_{PD}[n] = \begin{cases}
\tan^{-1}\left(\dfrac{IN_{imag}}{IN_{real}}\right) - \dfrac{\pi}{4}, & \dfrac{\pi}{2} > \theta_{in} \geqq 0 \\
\tan^{-1}\left(\dfrac{IN_{imag}}{IN_{real}}\right) - \dfrac{3\pi}{4}, & \pi > \theta_{in} \geqq \dfrac{\pi}{2} \\
\tan^{-1}\left(\dfrac{IN_{imag}}{IN_{real}}\right) + \dfrac{3\pi}{4}, & -\pi \leqq \theta_{in} < -\dfrac{\pi}{2} \\
\tan^{-1}\left(\dfrac{IN_{imag}}{IN_{real}}\right) + \dfrac{\pi}{4}, & -\dfrac{\pi}{2} \leqq \theta_{in} < 0
\end{cases}
\tag{6c}
$$

QPSK入力時

図11を見ると，位相検波器の利得は$K_{PD} = 1$（rad/rad）となります．

4）ループ・フィルタ

ループ・フィルタはPLL全体の特性（応答時間，周波数特性など）を決めます．その理由については，後ほど説明します．

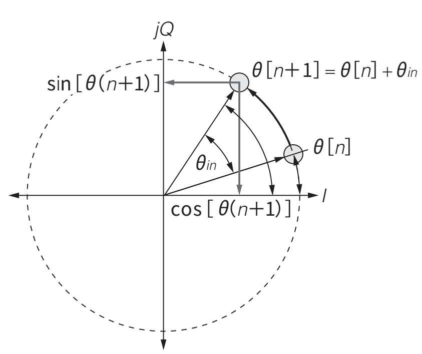


図10　NCOの動作

位相累算器によってθが徐々に大きくなるので，複素平面にある灰色の点はθ_{in}のステップで円を描く．灰色の点のI軸（実部）への投影はcos波形となり，Q軸（虚部）への投影はsin波形となる．

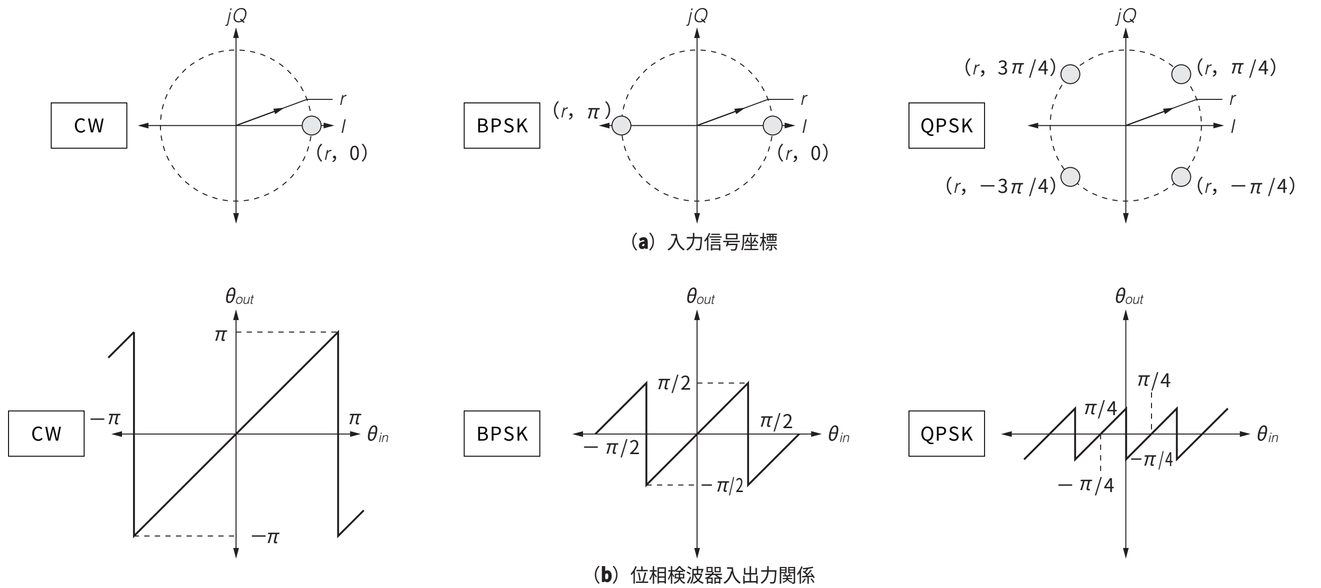


図11　入力信号と位相検波器入出力関係

(a)の座標はPLLが安定した状態を示す．QPSKの場合，π/4のオフセットがあるため，そのときの位相検波器の出力はゼロになる必要がある．これにより，データによる位相変更はなくなる．CWはcontinuous wave(連続波)の略である．BPSK(binary phase shift keying)には180°の位相不確定がある．つまり，信号が180°回転しても，見かけでは区別できない．この位相不確定は，PLLだけでは解決できない．判定部で調整する必要がある．

5) 遅延

全ディジタルPLLは帰還ループを組んでいるため，すべての出力を瞬時に知ることはできません．**図12**から，位相誤差のθ_eを計算するためには，あらかじめθ_iとθ_oを知らなければなりません．しかし，NCOの出力であるθ_oを計算するためにはθ_eを知らなければなりません．どちらが先か，鶏か，卵かといった問題です．このデッドロック(行き詰まり)を解決するために，遅延ブロックが必要です．つまり，遅延ブロックの初期値をゼロにすれば初期サンプルでは$\theta_e = \theta_i$となり，θ_oを知らなくても計算を進められます．各ブロックの出力を順番に計算していき，θ_oの計算結果は次のサイクルに使います．

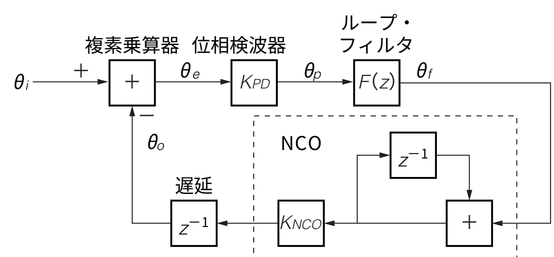


図12　PLLの直線化モデル

全ディジタルPLLでは入力信号の振幅が位相検波器で正規化されるため，動作に影響を与えない．複素乗算器では位相が加算されるはずだが，負帰還を組んでいるため，本モデルでは減算によってモデル化している(PLLを実装するとき，通常，ループ・フィルタの出力に−1を掛ける)．ここでは位相にしか注目していないため，NCOのsinとcosを作る部分を省いている．

● PLLの直線化モデルで考える

位相誤差が小さいとき，PLLは**図12**のように表現できます．

位相検波器では振幅が正規化されるので，PLLの直線化モデルでは振幅の影響を無視できます．複素乗算器の出力は次のようになります．

$$\theta_e(z) = \theta_i(z) - \theta_o(z) \qquad (7)$$

位相誤差は小さいので，位相検波器の出力に折り返しは起こりません．

$$\begin{aligned}\theta_p(z) &= K_{PD}\theta_e(z) \\ &= K_{PD}\left[\theta_i(z) - \theta_o(z)\right]\end{aligned} \qquad (8)$$

ループ・フィルタの出力は次のようになります．この時点では，ループ・フィルタの伝達関数$F(z)$は代表的な関数で示します．

$$\begin{aligned}\theta_f(z) &= F(z)\theta_p(z) \\ &= K_{PD}F(z)\left[\theta_i(z) - \theta_o(z)\right]\end{aligned} \qquad (9)$$

NCOの出力は次のようになります．位相だけに注目しているので，sinとcosの部分を省いています．

$$\theta_o(z) = \frac{K_{NCO}\theta_f(z)}{z-1} = \frac{K_{PD}K_{NCO}F(z)\left[\theta_i(z)-\theta_o(z)\right]}{z-1} = \frac{K_{PD}K_{NCO}F(z)}{z+[K_{PD}K_{NCO}F(z)-1]}\theta_i(z) \quad (10)$$

PLLの位相伝達関数は，次のように定義します．

$$H(z) = \frac{\theta_o(z)}{\theta_i(z)} = \frac{K_{PD}K_{NCO}F(z)}{z+\left[K_{PD}K_{NCO}F(z)-1\right]} \quad (11)$$

PLLの誤差伝達関数は，次のように定義します．

$$H_e(z) = \frac{\theta_e(z)}{\theta_i(z)} = \frac{\theta_i(z)-\theta_o(z)}{\theta_i(z)} = 1-H(z) = \frac{z-1}{z+\left[K_{PD}K_{NCO}F(z)-1\right]} \quad (12)$$

PLLは，用途によってはさまざまな信号変化に耐えなければなりません．基本的な信号変化には，次のようなものがあります．

- **ケース1：位相ステップ**――位相が急激に変化する（BPSK，QPSKなど）

$$\theta_1(z) = \Delta\phi\frac{z}{z-1} \quad (13a)$$

- **ケース2：周波数ステップ**――周波数が急激に変化する（チャネル切り替え，周波数ホッピングなど）

$$\theta_2(z) = \Delta\omega\frac{z}{(z-1)^2} \quad (13b)$$

- **ケース3：周波数ランプ**――周波数が徐々に変化する（ドップラ・シフトなど）

$$\theta_3(z) = \Delta\dot{\omega}\frac{z(z+1)}{(z-1)^3} \quad (13c)$$

PLLは，位相誤差 θ_e がゼロになるように動作します．位相誤差が安定した状態を数学的に表すと，次のようになります．

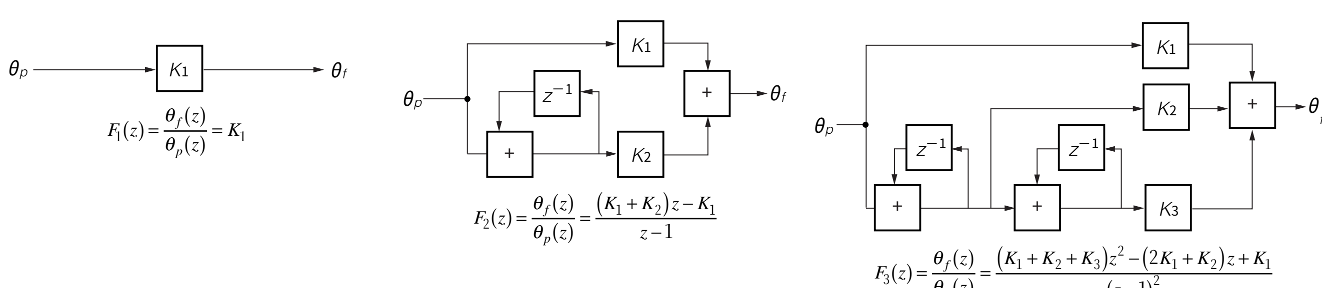
$$\lim_{n\to\infty}\theta_e[n] = \lim_{z\to 1}\theta_e[z]\frac{z-1}{z} = \lim_{z\to 1}\frac{\theta_i(z)(z-1)^2}{z\left\{z+\left[K_{PD}K_{NCO}F(z)-1\right]\right\}} \quad (14)$$

ケース1：$\theta_i(z) = \theta_1(z)$ のとき，

$$\lim_{n\to\infty}\theta_{e1}[n] = \lim_{z\to 1}\Delta\phi\frac{z}{z-1}\frac{(z-1)^2}{z\left\{z+\left[K_{PD}K_{NCO}F(z)-1\right]\right\}} = \lim_{z\to 1}\Delta\phi\frac{z-1}{z+\left[K_{PD}K_{NCO}F(z)-1\right]} = 0,\quad F(z=1)\neq 0 \quad (15)$$

ループ・フィルタの伝達関数は $F(z) = K_1$（固定値）でこの条件を満足します（**図13(a)**）．

ケース2：$\theta_i(z) = \theta_2(z)$ のとき，



(**a**) 0次ループ・フィルタ（位相ステップ）
(**b**) 1次ループ・フィルタ（周波数ステップ）
(**c**) 2次ループ・フィルタ（周波数ランプ）

図13　ループ・フィルタの構成

入力信号が位相ステップのとき，位相誤差がゼロになるように，(a)のような構成を利用できる．ただし，入力信号が周波数ステップの場合，この構成では位相誤差をゼロにできない．そこで，入力信号が周波数ステップの場合は，(b)のような構成を利用する．入力信号が周波数ランプの場合は，この構成では位相誤差をゼロにできない．そこで，(c)のような構成を利用する．

$$\lim_{n\to\infty}\theta_{e2}[n]=\lim_{z\to1}\Delta\omega\frac{z}{(z-1)^2}\frac{(z-1)^2}{z\left\{z+\left[K_{PD}K_{NCO}F(z)-1\right]\right\}}$$
$$=\lim_{z\to1}\Delta\omega\frac{1}{z+\left[K_{PD}K_{NCO}F(z)-1\right]} \qquad (16)$$

ループ・フィルタが$z=1$で次数mのポール（極）を持つとき，その伝達関数は次のように表現できます．

$$F(z)=\frac{F_O(z)}{(z-1)^m} \qquad (17)$$

そうなると，

$$\lim_{n\to\infty}\theta_{e2}[n]=\lim_{z\to1}\Delta\omega\frac{1}{z+\left[K_{PD}K_{NCO}\dfrac{F_O(z)}{(z-1)^m}-1\right]}$$
$$=\lim_{z\to1}\Delta\omega\frac{1}{z+\left[\dfrac{K_{PD}K_{NCO}F_O(z)-(z-1)^m}{(z-1)^m}\right]}$$
$$=\lim_{z\to1}\Delta\omega\frac{(z-1)^m}{K_{PD}K_{NCO}F_O(z)}$$
$$=0,\quad m\geqq1,\quad F_O(z=1)\neq0 \qquad (18)$$

この条件を満たすループ・フィルタの構成を**図13（b）**に示します．同じように，ケース3の周波数ランプに対するループ・フィルタの構成を**図13（c）**に示します．

1次ループの伝達関数の場合，次のようになります[注6]．

$$H_1(z)=\frac{K_{PD}K_{NCO}F_1(z)}{z+\left[K_{PD}K_{NCO}F_1(z)-1\right]}$$
$$=\frac{K_{PD}K_{NCO}K_1}{z+\left[K_{PD}K_{NCO}K_1-1\right]} \qquad (19)$$
$$=\frac{\alpha}{z+(\alpha-1)},\quad \alpha=K_{PD}K_{NCO}K_1$$

2次ループの場合，次のようになります．

$$H_2(z)=\frac{K_{PD}K_{NCO}F_2(z)}{z+\left[K_{PD}K_{NCO}F_2(z)-1\right]}$$
$$=\frac{K_{PD}K_{NCO}\left[\dfrac{(K_1+K_2)z-K_1}{z-1}\right]}{z+\left\{K_{PD}K_{NCO}\left[\dfrac{(K_1+K_2)z-K_1}{z-1}\right]-1\right\}}$$
$$=\frac{K_{PD}K_{NCO}\left[(K_1+K_2)z-K_1\right]}{z(z-1)+K_{PD}K_{NCO}\left[(K_1+K_2)z-K_1\right]-(z-1)}$$
$$=\frac{K_{PD}K_{NCO}(K_1+K_2)z-K_{PD}K_{NCO}K_1}{z^2+\left[K_{PD}K_{NCO}(K_1+K_2)-2\right]z+\left(1-K_{PD}K_{NCO}K_1\right)}$$
$$=\frac{(\alpha+\beta)z-\alpha}{z^2+(\alpha+\beta-2)z+(1-\alpha)},\quad \alpha=K_{PD}K_{NCO}K_1,\quad \beta=K_{PD}K_{NCO}K_2 \qquad (20)$$

周波数応答$H(e^{j2\pi fT})$は，$z=\exp(j2\pi fT)$を代入することによって得られます．$|H(e^{j2\pi fT})|^2$を計算すると，周波数応答の振幅特性が得られます．

ループ・ノイズ帯域幅（B_L）は，システムが通すノイズ電力と等価なノイズ電力を通す矩形フィルタの通過周波数の値です（**図14**）．数学的に表すと，次のようになります．

$$2B_L=\int_{-1/2T}^{1/2T}\left|H\left(e^{j2\pi fT}\right)\right|^2df \qquad (21)$$

ループ・フィルタのパラメータを計算するとき，このB_Lを決めてからほかのパラメータを算出します．一般に，$B_{L(W)}=0.1T$のときをワイド（広帯域），$B_{L(M)}=0.05T$のときをミディアム（中帯域），$B_{L(N)}=0.01T$のときをナロー（狭帯域）と言います．ワイドのときは捕捉範囲が広いので，周波数誤差を吸収します．しかし，このモードではノイズもたくさん通過するので，ノイズに弱いと言えます．つまり，安定状態からよく外れます．

ループ・ノイズ帯域幅が狭いとノイズに強くなりますが，周波数捕捉範囲は狭くなります．捕捉するときはB_Lを広くし，捕捉してから徐々にB_Lを狭くするのが普通の使いかた

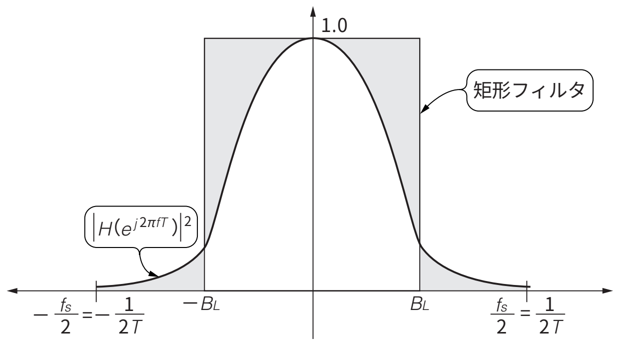


図14 ループ・ノイズ帯域幅
システムが通すノイズ電力は$|H(e^{j2\pi fT})|^2$の面積．矩形フィルタが通すノイズ電力は$2B_L$．両方の面積が同じになるときのB_Lはループ・ノイズ帯域幅となる．等価ノイズ帯域幅とも呼ばれる．

注6：PLL全体の次数はループ・フィルタの次数＋1になる（NCOが存在するため）．

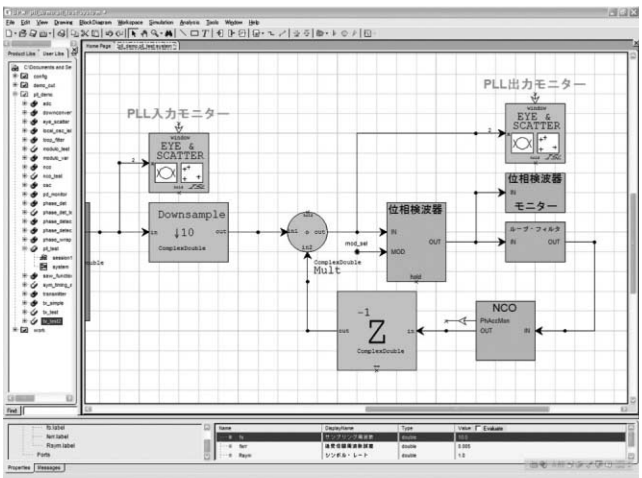


図15　PLL全体を組んだシステム
さまざまなパラメータを変更して，動作や反応がどう変わるかを確かめていただきたい．

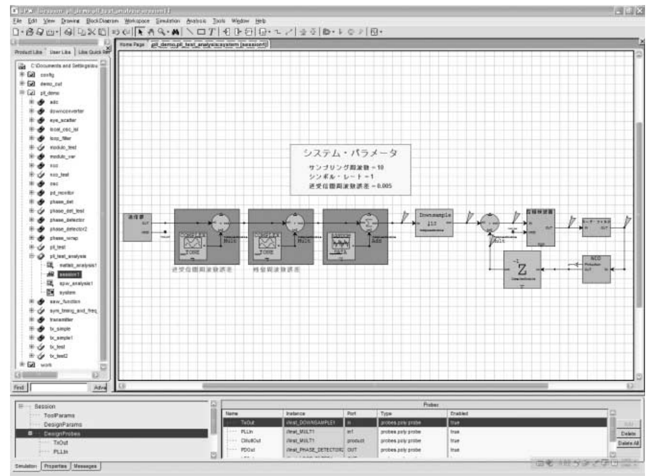


図16　解析用のシステム
実行するサンプル数は，Simulationタブの「Session」→「ToolParams」を選択することで設定できる．観測したい信号にプローブを付けて，実行後，アナリシスGUIで確認する．

です．

PLL設計は複雑な分野なので，この記事の中ですべてを紹介することはできません．詳しくは参考文献(2)，(3)を参照してください．PLL全体を組んだシステムはpll_test.systemに用意されています(**図15**)．PLLの動作を検証するため，このデータを実行してみてください．

● 信号をファイルに保存して解析

ここまで説明したデモンストレーション・システムでは，PLLの動作を理解しやすいように，シミュレータが備えているISL(interactive simulation library)という機能を利用しました．ISLは，値をスライダで対話的に変更したり，測定器のように信号を表示するためのライブラリです．一方，瞬時の動きを確認したり，後処理を行いたい場合は，信号をファイルに保存したほうが便利です．

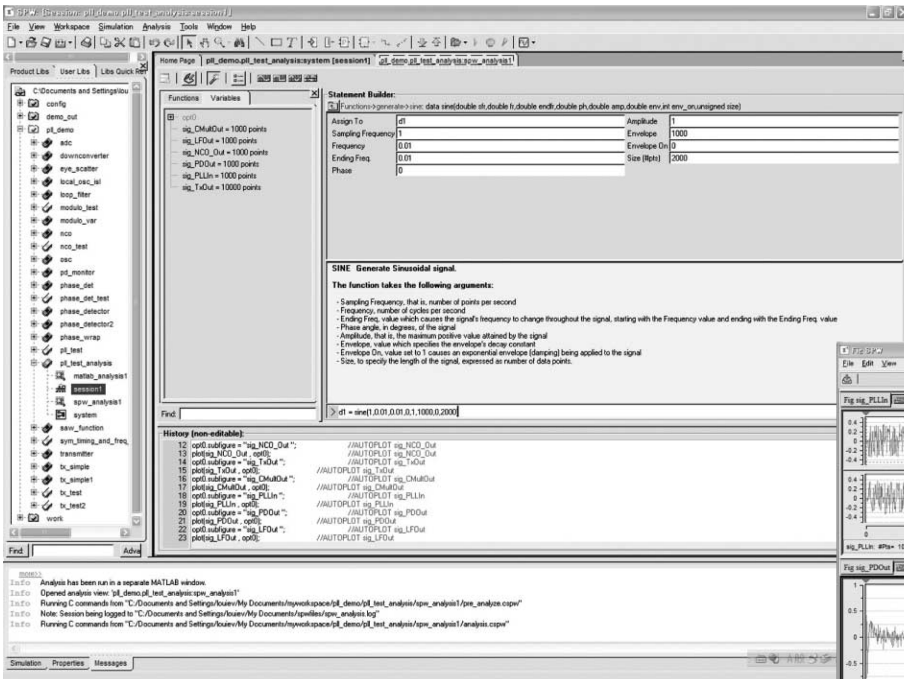
信号をファイルに保存する際のサンプル数を設定する場合は，「Simulation」→「Open Session」を実行します(システムを最初に実行するとき，自動的にsessionが生成される)．下にある情報表示領域にSimulationタブが表示されます．ここで，ToolParamsの「Length of Simulation (samples)」の値を設定します．なお，この入力フィールドに「Indefinite」と入力すると，実行可能な最大サンプル数の信号が保存されます．

User Libsからpll_demoの下にある「pll_test_analysis.session1」を開いて実行してみてください(**図16**)．systemとsessionは基本的に同じですが，実行するための情報はsessionに含まれています．これは，一つのシステムに対して複数のテストベンチを適用したいときなどに便利な機能です．

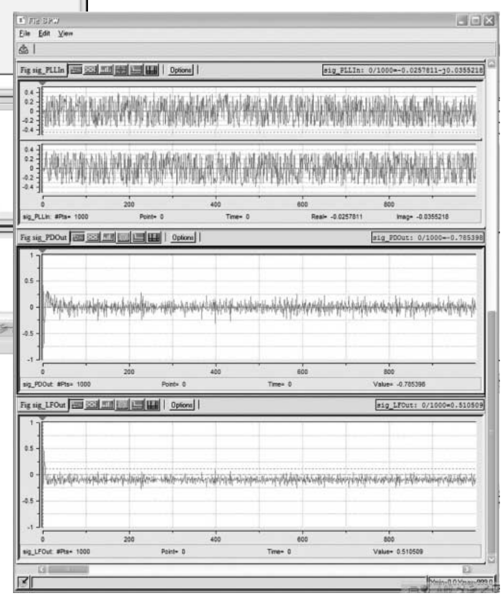
Sessionで信号をプローブするとき，まず観測したいポイントをマウスで選択します．信号が選択されると複数の小さい黄色い箱がそのワイヤを囲みます．メニュー・バーのSTOPアイコンの右に，緑色のプローブ・アイコンがあります．これをクリックすると，プローブを接続できます．また，Simulationタブの「Design Probes」を選択すると，右にプローブ表が現れます．Nameの欄をダブルクリックすると，名まえを変更できます．

このsessionを実行すると，設定したサンプル数分のシミュレーションを実行します．ウィンドウの右下にプログレス・バーが現れ，100%になるとシミュレーションが終了します．プローブした信号を見るために，メニュー・バーのいちばん右にあるアナリシス・アイコンをクリックしてください．クリックすると，「アナリシスGUI(Analysis GUI)」と呼ぶ画面が立ち上がります(**図17(a)**)．プローブした信号はプロット・ウィンドウに表示されます(**図17(b)**)．

アナリシスGUIでは，プローブした信号は変数のように取り扱うことができます．すなわち，複数の信号を足したり，引いたりすることができます．詳細はオンライン・ドキュメントの「Basic SPW 5-XP Tutorial」を参照してください．



(a) アナリシスGUI



(b) プロット・ウィンドウ

図17 解析用の画面

(a)では，デザイン領域(画面の中央から右端までの領域)の左側に変数と関数の一覧が，右側にStatement Builderとそのヘルプが表示されている．また，そのすぐ下にはコマンド・ラインの領域が，さらにその下に履歴(ヒストリ)の表示領域がある．解析用のコマンドは，コマンド・ラインに入力する．デフォルトでは，プローブした信号が自動的にプロットされる．(b)のプロット・ウィンドウの波形の上にあるアイコン群によって，信号のアイ・パターンや棒グラフ，座標，周波数特性(FFT)，数値などの表示を切り替える．信号の上にある三角形は信号カーサで，これが指している信号点の値は波形の右上に表示される．

● 最適化された完全なPLLの設計にチャレンジ！

ここではPLLの動作と設計の過程を簡単に説明しました．ここで示したPLLのデモンストレーション・システムは最適化されていません．周波数を急に変更すると，PLLは不安定な状態になります．PLLに興味のある方は，ぜひ完全なPLLになるように設計に手を入れてみてください．

なお，本稿ではSPW 5-XPの一部の機能しか使用していません．コーウェアのホームページに本ツールのアプリケーション・ノートや新しい情報が公開されているので，参考にしてください．

参考・引用*文献

(1) 遠坂俊昭；PLL回路の設計と応用，CQ出版社，2003年．
(2) Heinrich Meyr and Gerd Ascheid；Synchronization in Digital Communications, Vol. 1: Phase-, Frequency- Locked Loops, and Amplitude Control，John Wiley & Sons, Inc.，1990.
(3) S. A. Stephens and J. B. Thomas；Controlled-Root Formulation for Digital Phase-Locked Loops，IEEE Transactions on Aerospace and Electronic Systems，vol.31，no.1，Jan. 1995.
(4) コーウェアのホームページ，http://www.coware.co.jp/

ルーウィ・ヴァレニャ
コーウェア(株)

<筆者プロフィール>

T. Louie Valeña．1987年，フィリピンのUniversity of the Philippines電気工学部を卒業し，Mitech社に入社．テレックス・モデムの開発・設計に従事．1990年，日放電子に入社．ディジタル通信システムの開発・設計に従事．DSPアルゴリズムとソフトウェア開発，ディジタル/アナログ・RF回路の設計を担当．1995年，米国University of Southern California大学院電気工学部修士課程を修了し，修士号を取得．2001年，日本ケイデンスデザインシステムズ社に入社．SPWのアプリケーション・エンジニア(AE)としてプリセールス活動に従事．現在，コーウェアでSPWのAEとして活躍している．